A Comparative Study of Probability Collectives Based Multi-agent Systems and Genetic Algorithms

ABSTRACT

We compare Genetic Algorithms (GA's) with Probability Collectives (PC), a new framework for distributed optimization and control. In contrast to GA's, PC-based methods do not update populations of solutions. Instead they update an explicitly parameterized probability distribution p over the space of solutions. That updating of p arises as the optimization of a functional of p. The functional is chosen so that any p that optimizes it should be p peaked about good solutions. The PC approach works in both continuous and discrete problems. It does not suffer from the resolution limitation of the finite bit length encoding of parameters into GA alleles. It also has deep connections with both game theory and statistical physics. We review the PC approach using its motivation as the information theoretic formulation of bounded rationality for multi-agent systems. It is then compared with GA's on a diverse set of problems. To handle high dimensional surfaces, in the PC method investigated here p is restricted to a product distribution. Each distribution in that product is controlled by a separate agent. The test functions were selected for their difficulty using either traditional gradient descent or genetic algorithms. On those functions the PC-based approach significantly outperforms traditional GA's in both rate of descent, trapping in false minima, and long term optimization.

Categories and Subject Descriptors

[**Estimation of distribution algorithms**]

Keywords

Multi-agent systems, Probability Collectives, Product Distribution theory, genetic algorithms, estimation of distribution algorithms

1. INTRODUCTION

Genetic algorithms (GA) [9] have been used as computational models of natural evolutionary systems and as adaptive algorithms for solving complex problems. At the core of this type of optimization algorithm is a population of solutions that are used to attempt to solve the problem at hand, in contrast to single-solution based algorithms such as Simulated Annealing and Extremal Optimization. In the past decade, the research on Multiagent Systems (MAS) has led to an advanced class of population-based adaptive algorithms that aims to provide both principles for construction of complex systems involving multiple agents and mechanisms for coordination of interacting agents' behavior. While there is no generally accepted definition of "agent" in Computer Science [17], for the purposes of this paper, we consider an agent to be an entity, such as a robot, which is self-interested and capable of learning in an environment. Furthermore, there is a specified system objective with respect to MAS which rates the performance of the joint actions of the agents.

Typically the search of adaptive, distributed agent-based algorithms is conducted by having each agent run its own reinforcement learning algorithm [29, 26, 20]. In this methodology the global utility function $G(x)$ in the system maps a joint move of the agents, $x \in X$, to the performance of the overall system. However, in practice the agents in a MAS are bounded rational; the equilibrium they reach typically involves mixed strategies rather than pure strategies – i.e., they don't settle on a single point x optimizing $G(x)$. This suggests formulating a framework to explicitly account for the bounded rational, mixed strategy character of the agents. Probability Collectives (PC) adopts this perspective to show that the equilibrium of a MAS is the minimizer of a Lagrangian $\mathcal{L}(P)$ (derived using information theory) that quantifies the expected value of G for the joint distribution $P(x_1, x_2, ..., x_N)$ [25, 23, 22].

Now consider a bounded rational game in which the agents are independent, with each agent i choosing its move x_i at any instant by sampling its probability distribution (mixed strategy) at that instant, $q_i(x_i)$. Accordingly, the probability distribution of the joint-moves is a *product distribution*; i.e., $P(x) = P(x_1, x_2, ..., x_N) = \prod_{i=1}^{N} q_i(x_i)$, if there are N agents participate in the game. In this representation of a MAS, lacking the full joint probability distribution, all coupling between the agents occurs indirectly. It is the separate distributions of the agents $\{q_i\}$ that are statistically coupled, while the actual moves of the agents are independent.

The core of PC-based algorithms is thus to approximate the joint distribution by the product distribution, and to concentrate on how the agents update the probability distributions across their possible actions instead of specifically on

WOODSTOCK GECCO 2005, Washington DC, USA

the joint action generated by sampling those distributions.

The PC approach differs from traditional optimization methods such as gradient descent or GA which concentrate on a specific choice for the design variables (i.e. pure strategies) and on how to update that choice. Since the PC approach operates directly on probability distributions, it offers a direct treatment for incorporating uncertainty, which is also represented through probabilities [3]. This is the most salient feature that this class of algorithms possesses – the search course is guided by a probability distribution over x, rather than a single value of x. By building such a probabilistic model of promising solutions and sampling the built model to generate new candidate solutions, PC allows the agents to significantly expand the range of exploration of the search space, and simultaneously focus on promising solutions areas. As a result, the estimation of distribution algorithms can provide a robust and scalable solution to many important classes of optimization problems.

2. REVIEW OF RELATED WORK

Many techniques in the evolutionary computation community use Boltzmann distributions in ad hoc ways to update a "population" of $\mathbf{x}$'s, e.g., "truncation selection" and "Boltzmann selection." These rules are not formally derived, and do not directly concern themselves with distributions q. However, some of them are similar to the PC update rules, in particular iterative focusing rules, only applied to sets of Monte Carlo sample points (the population) rather than to q. There are other techniques which also use Boltzmann distributions, although without a population, e.g., simulated annealing.

These early techniques do not consider the underlying distribution that gets sampled to produce the population. Such consideration was introduced in PBIL [2], MIMIC [4] and other Estimation of Distribution Algorithms (EDAs) [11], followed shortly by the powerful CE method [16]. EDAs replace traditional variation operators of genetic and evolutionary algorithms, such as mutation and crossover, by building a probabilistic model of promising solutions and sampling the built model to generate new candidate solutions.

However while considering distributions, none of this early work casts the objective as a minimization of a functional of that distribution. Accordingly, all the power arising from minimizing Euclidean vectors is absent in this work. There is none of the second order methods, difference utilities, or data-ageing that appear to be crucial for very large problems.

There is other previous work on optimization that has directly considered the distribution q as the object of interest such as deterministic annealing [5] when the conditional G can be evaluated in closed form, and therefore has no concern for Monte Carlo sampling issues. Most tantalizingly, Probability matching [18] uses Monte Carlo sampling to optimize a functional of q. However this work was in the context of a single agent, did not exploit the vector space properties, and was not pursued.

Other work has both viewed q as the fundamental object of interest and used techniques like data-aging and difference utilities [28, 27, 24]. However this work was not based on information-theoretic considerations and had no explicit objective function for q. It was the introduction of such considerations that resulted in PC. Finally, shortly after the introduction of PC a variant of its Monte Carlo updating has been introduced, called the MCE method [15].

3. BOUNDED RATIONAL GAME THEORY

In this section we review PC as the information-theoretic formulation of bounded rational game theory [3, 12].

3.1 Review of noncooperative game theory

Assume that a set of N players participate in a noncooperative game. Each player i has its own set of allowed *pure strategies*. A *mixed strategy* is a distribution $q_i(x_i)$ over player i's possible pure strategies [6]. Each player i also has a private utility function g_i that maps the pure strategies adopted by all N of the players into the real numbers. So given mixed strategies of all the players, the expected utility of player i is

$$E(g_i) = \int dx \prod_j q_j(x_j) g_i(x)$$

In a Nash equilibrium, every player adopts the mixed strategy that maximizes its expected utility, given the mixed strategies of the other players. Nash equilibria require the assumption of full rationality, that is, every player i can calculate the strategies of the other players and its own associated optimal distribution.

3.2 Review of the maximum entropy principle

In the absence of full rationality, the equilibrium is determined based on the information available to the players. The Shannon entropy, $S(P) = -\int dy P(x) \ln(P(x))$, is a unique real-valued quantification of the amount of syntactic information in a distribution $P(x)$. Hence, the distribution with minimal information is the one that does not distinguish at all between the various x, i.e., the uniform distribution. Given some incomplete prior knowledge about a distribution $P(x)$, this says that the estimate $P(x)$ should contain the minimal amount of extra information beyond that already contained in the prior knowledge about $P(x)$. This approach is called the maximum entropy (maxent) principle and it has proven useful in domains ranging from signal processing to supervised learning [13].

3.3 Maxent Lagrangians

One can consider an external observer of a multi-player game attempting to determine the equilibrium, i.e., the joint strategy that will be followed by real-world players of the game. The observer only knows the expected utility for each player rather than the full joint probability distribution the players are following. The best estimate of the joint distribution q that generated those expected utility values, by the maxent principle, is the distribution with maximal entropy, subject to those expectation values.

For simplicity assume a finite number of players and a discrete set of possible strategies for each player. In addition, to agree with convention in other fields, the sign of each g_i is flipped so that the associated player i wants to minimize that function rather than maximize it. (Intuitively, this flipped g_i can be regarded as the "cost" to player i.)

For prior knowledge consisting of the set of expected utilities of the players $\{\epsilon_i\}$, the maxent estimate of the associated q is given by the minimizer of the Lagrangian:

$$\mathcal{L}_q \equiv \sum_i \beta_i [E_q(g_i) - \epsilon_i] - S(q)$$
$$= \sum_i \beta_i [\int dx \prod_j q_j(x_j) g_i(x) - \epsilon_i] - S(q), \quad (1)$$

where the subscript on the expectation value indicates that it is evaluated under distribution q, and the $\{\beta_i\}$ are "inverse temperatures" (i.e., $\beta_i = 1/T_i$) implicitly set by the constraints on the expected utilities.

The mixed strategies minimizing the Lagrangian are related to each other via

$$q_i(x_i) \propto e^{-E_{q_{(i)}}[G|x_i]}, \quad (2)$$

where the overall proportionality constant for each i is set by normalization, and

$$G(x) \equiv \sum_i \beta_i g_i(x).$$

The subscript $q_{(i)}$ on the expectation value indicates that it is evaluated according to the distribution $\prod_{j \neq i} q_j$. The expectation is conditioned on player i making move x_i. Eq. (2) shows that the probability of player i choosing pure strategy x_i depends on the effect of that choice on the utilities of the other players.

Now consider the case of maximal prior knowledge for the behavior of player i. Here the actual joint-strategy of the players and therefore all of their expected utilities are known. For this case, trivially, the maxent principle says the "estimate" q is that joint-strategy (it being the q with maximal entropy that is consistent with the prior knowledge). The same conclusion holds if the prior knowledge also includes the expected utility of player i.

Removing player i's strategy from this maximal prior knowledge leaves the mixed strategies of all players other than i, together with player i's expected utility. Now the prior knowledge of the other players' mixed strategies can be directly incorporated into a maxent Lagrangian for each player:

$$\mathcal{L}_i(q_i) \equiv \beta_i [\epsilon_i - E(g_i)] - S_i(q_i)$$
$$= \beta_i [\epsilon_i - \int dx \prod_j q_j(x_j) g_i(x)] - S_i(q_i).$$

The solution is a set of coupled Boltzmann distributions:

$$q_i(x_i) \propto e^{-\beta_i E_{q_{(i)}}[g_i|x_i]}. \quad (3)$$

Following Nash, Brouwer's fixed point theorem can be used to establish that for any non-negative values $\{\beta\}$, there must exist at least one product distribution given by the product of these Boltzmann distributions (one term in the product for each i).

The first term in $\mathcal{L}_i$ is minimized by a perfectly rational player. The second term is minimized by a perfectly irrational player, i.e., by a perfectly uniform mixed strategy q_i. Thus β_i in the maxent Lagrangian explicitly specifies the balance between the rational and irrational behavior of the player. When $\beta \to \infty$, the set of q that simultaneously minimize the Lagrangians will recover the Nash equilibria of the game, which is the set of delta functions about the Nash equilibria. The same is true for Eq. (2). In fact, Eq. (2) is just a special case of Eq. (3), where all player's share the same private utility, G. Such games are known as *team games*. This reflects the fact that for this case, the difference between the maxent Lagrangian and the one in Eq. (1) is independent of q_i. Due to this relationship, the guarantee of the existence of a solution to the set of maxent Lagrangians implies the existence of a solution of the form for Eq. (2).

3.4 Optimizing the Lagrangian

Given that the agents in a multi-agent system are bounded rational, if they play a team game with world utility G, their equilibrium will be the optimizer of G. Furthermore, if constraints are included, the equilibrium will be the optimizer of G subject to the constraints. The equilibrium can be found by minimizing the Lagrangian in Eq. (1) where the prior information set is empty, e.g. for all i, $\epsilon_i = \{0\}$.

Specifically for the unconstrained optimization problem,

$$\min_{\vec{x}} G(\vec{x})$$

assume each agent sets one component of $\vec{x}$ as that agent's action. The Lagrangian $\mathcal{L}_i(q_i)$ for each agent as a function of the probability distribution across its actions is,

$$\mathcal{L}_i(q_i) = E[G(x_i, x_{(i)})] - TS(q_i)$$
$$= \sum_{x_i} q_i(x_i) E[G(x_i, x_{(i)})|x_i] - TS(q_i),$$

where G is the world utility (system objective) which depends upon the action of agent i, x_i, and the actions of the other agents, $x_{(i)}$. The expectation $E[G(x_i, x_{(i)})|x_i]$ is evaluated according to the distributions of the agents other than i:

$$P(x_{(i)}) = \prod_{j \neq i} q_j(x_j).$$

The entropy S is given by:

$$S(q_i) = -\sum_{x_j} q_i(x_j) \ln(q_i(x_j)).$$

Each agent then addresses the following local optimization problem,

$$\min_{q_i} \mathcal{L}_i(q_i),$$
$$\text{s. t. } \sum_{x_i} q_i(x_i) = 1, \; q_i(x_i) \geq 0, \; \forall x_i.$$

During the minimization of the Lagrangian, the temperature T offers a means to adjust the degree of the exploitation of existing promising solutions (low temperature) and that of the exploration of the search space (high temperature).

One can employ gradient descent or Newton updating to minimize the Lagrangian since both the gradient and the Hessian are obtained in closed form. Using Newton updating and enforcing the constraint on total probability, the following update rule at each iteration is obtained [21]:

$$q_i(x_i) \to q_i(x_i) - \alpha q_i(x_i) \times$$
$$\{(E[G|x_i] - E[G])/T + S(q_i) + \ln q_i(x_i)\},$$

where α plays the role of a step size. The step size is required since the expectations result from the current probability distributions of all the agents. The update rule ensures that the total probability sums to unity but does not prevent negative probabilities. To ensure this, all negative components are set to a small positive value, typically 1×10^{-6}, and then the probability distribution is re-normalized.

3.5 Product Distribution MAS Algorithms

To perform this gradient descent in probability space each agent must estimate the expected value of any of its actions, $E[G|x_i]$, from monte-carlo samples. The optimization for both the discrete and continous parameter optimization was discussed in Bieniawski, Kroo & Wolpert [3]. Briefly, optimization proceeds in alternating rounds of monte-carlo sampling blocks, and updates to the agents's probability distribution over the parameter value. To draw a monte-carlo sample each agent chooses the value for its parameter x_i from its current probability distribution, and the world cost function $G(x)$ is evaluated. We note that although we use a world cost function for pedagogical simplicity, other formulations in which each agent has its own cost function may be more desirable in under-sampled problems.

Each agent computes an estimate of the expected cost as a function of the parameter value, $E[G|x_i]$, by interpolation over the continuous range from the sampled values using a Gaussian kernel density estimator (i.e. a convolution over the delta function samples). To permit sampling from this implicit probability distribution (PD) over the continuous range, it is evaluated on a grid and the probability of any off-grid points is linearly interpolated.

The number of samples in each monte-carlo block determines accuracy of the expected cost estimate. Here we assume the case that sampling the objective function is costly, so we wish to gain the most information from the least number of samples. The kernel density estimation implies and exploits weak prior knowledge about smooth interpolation between the sample points. Additionally, as long as each iteration update does not dramatically change the PD we can re-use samples from the previous iterations, geometrically weighting them according to their "age" in interations. One can crudely consider the imperfections that these augmentations introduce as another contribution to the bounded rationality term that broadens the probability distribution.

The primary free parameters in the optimization are the gaussian kernel width (τ), the rate of cooling ($\delta T/T$), the number of monte-carlo samples per iteration, the proportional step size in the gradient descent (α), and data-aging rate (γ). Since the cooling is geometric the initial temperature T selection is logarithmically insensitive, and in practice can be adaptively set after observing the domain of the objective function values. The grid size of the PD representation is an asymptotically unimportant free parameter trading numerical precision for computation time.

4. EXPERIMENTAL RESULTS

In this section, we report a comparison of a Multi-agent Probability Collective (see [3] for the detailed algorithm) with a Genetic algorithm in searching for the global minimum of four traditional test function surfaces of increasing complexity and difficulty. Many of the characteristics in these testbeds are considered important by evolutionary algorithm practitioners, such as multimodality, nonlinearity and non-separability, etc. The study of search efficiency usually involves defining a performance measure that embodies the idea of rate of improvement, so that its change over time can be monitored for investigation. In many practical problems, a traditional performance metric is the "best-so-far" curve that plots the fitness of the best individual that has been seen thus far by generation n for the GA, i.e., a point in the search space that optimizes the objective function thus far. The best-so-far curves presented for each testbed are the mean over 50 runs and error bars on the graph show the 95% confidence intervals about the mean.

In contrast, in the PC, the result returned is a probability distribution across the variable space that optimizes an associated Lagrangian. It embodies a notion of a region where the minimum is likely to be located as well as an uncertainty due to both imperfect sampling, and the stochastic independence of the agents' actions. To compare this on even footing to the GA, we focus solely on the samples themselves. We note however, this is an imperfect summary: for example, generally the maximum of the PD will not actually be one of the samples, though it is one's best single guess of the minimum.

In a basic genetic algorithm the population consists of genotypes that encode solutions (phenotypes) to some problems. Evolution occurs by iterated stochastic variation of genotypes, and selection of the best phenotypes is according to how well they optimize the function of interest. Table 1 depicts the process of a simple genetic algorithm by which we use to compare with the PD-based MAS algorithm.

Table 1: Mechanism of a simple GA.

1. Randomly generate an initial population of l n-bit genotypes (individuals).
2. Evaluate each individual's fitness.
3. Repeat until l offspring have been created.
a. select a pair of parents for mating;
b. apply crossover operator;
c. apply mutation operator.
4. Replace the current population with the new population.
5. Re-interate from Step 2 until terminating condition.

For our traditional GA, we examined a range of population sizes (50,100, 200 and 500) to bracket a range of initial descent rates and long term performance. The 200 member GAs generally had the same long-term performance but converged faster than the GAs with 500 members. The 50 member populations generally descended quickly but converged to sub-optimal solutions. Parameter values were finely discretized to approximate a continuous range, and encoded as bit strings. (Various bit lengths were tried before settling upon 20 or 50 bits.) The GA experiments employ a binary tournament selection [8], one-point crossover and mutation rates of 0.7/pair and 0.005/bit, respectively.

In the following examples the optimization free parameters for the PC were set as follows: step size $\alpha = 0.2$, data-ageing rate $\gamma = 0.5$, cooling rate $\delta_T/T = 0.01$, Gaussian kernel width τ is set to 1% of the range of the search parameter, and $T = 0.1$ was a sufficiently high starting temperature. Monte-calro block sizes of 50 and 25 were examined. Interestingly, using more samples per iteration did not significantly improve the best-so-far value for any given iteration. Thus only the results for the 25 monte-carlo blocks

are reported since they use fewer samples per iteration.

4.1 The Schaffer Function F_7

Schaffer's test function F_7 [19] is defined as:

$$f(\overline{x}) = (x_1^2 + x_2^2)^{0.25}[sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1],$$

where $-1 \leq x_i \leq 1$ for $1 \leq i \leq 2$. Figure 1.a displays the surface which is plotted upside down for easier viewing of the inverted minimum as a peak. Since there are many local optima in the search space, the population in the GA can easily converge on any of them. The barriers would also present considerable difficulty to search approaches that evolve a single point x using local gradient information.

For this simple 2-dimensional case one could feasibly model the probability distribution in the full joint PC space rather than approximating it as a product distribution. However since we are, in fact, exploring multi-agent systems, instead two agents will carry out the search in the two parameters independently. For the GA, each variable is encoded by 50 bits; thus each agent in the GA consists of a bit string of length 100 (two blocks of 50 bits are concatenated to form a string).

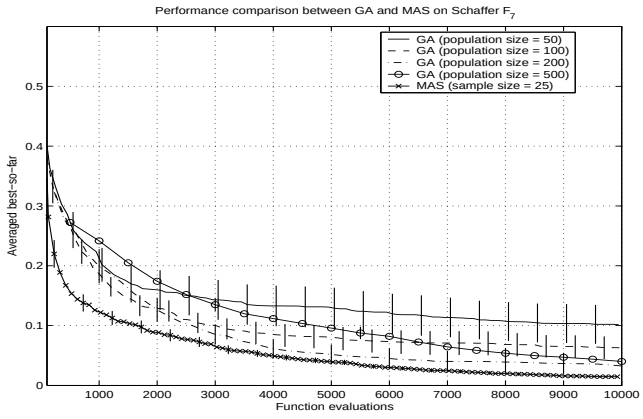

Figure 2: Best-so-far performance on Schaffer F_7.

Figure 2 displays the best-so-far values attained by the Multi-agents system(MAS) and the GA as a function of the number of sample evaluations of the objective function. The curves are the mean values over 50 repetitions and the vertical bars are the 95% confidence intervals on the means. Curves for different population sizes of the GA are shown. The methods distinguish themselves with different rates of initial descent of the objective function (on left) and the long-term performance (on right). Notably, the run-to-run variation of the performance trajectory is much lower on the PC-based MAS than for the GA (see vertical bars).

Figure 3 displays the evolution of the probability distribution of the two variables for a typical MAS run. As can be seen, the probability density quickly centers about the optimum. This explains why the PC-based MAS is able to locate the optimum in a rather short period of time.

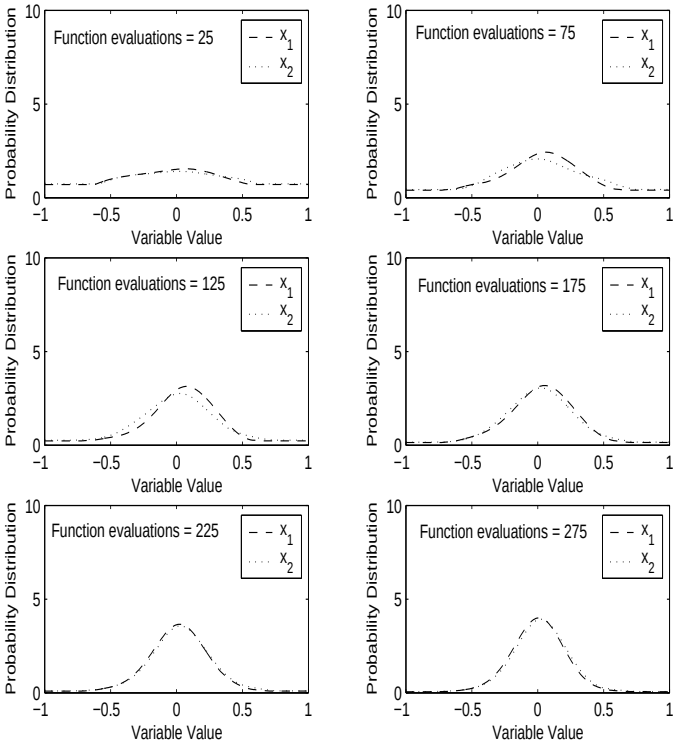

Figure 3: Evolution of probability distribution of x_1 and x_2 on Schaffer F_7.

4.2 The Rosenbrock Function

The second testbed is the generalized Rosenbrock function in ten dimensions. The definition of this function is [10]:

$$f(\overline{x}) = \sum_{i=1}^{N-1} [100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2], \qquad (4)$$

where $\overline{x} = [x_1, x_2, \ldots, x_N]^T$, $-5.12 \leq x_i \leq 5.12$.

Although the problem we use here is 10 dimensional (N=10) with ten agents, we provide the reader a visual gist of the surface by showing the Rosenbrock function in two-dimensions. Again the plot is reversed for easy viewing of the minimum as a peak. Since the spike is so sharp, a logarithmic vertical axis is displayed in Figure 1.b.

Rosenbrock's saddle is a classic optimization problem with a narrow global optimum hidden inside a long, narrow, curved flat valley. Monte-carlo methods will have difficulty landing a point in the narrow spike and thus will not efficiently locate it. The U-shape will also tend to make decomposition of the PC into a product distribution challenging. Since it has no barriers the surface would be ripe for gradient descent; however while the valley will be found quickly the curvature and flatness of the valley floor will frustrate sampled gradient estimation.

Each individual of the population of the GA is a 200 bit-string concatenated by ten blocks of 20 bits each encoding a variable. The empirical results are displayed in Figure 4. The top of this figure shows the best-so-far values attained by the algorithms. The bottom plot displays a detailed view from the range of the objective's value on the interval [0, 1000]. One can clearly see that the PC technique can locate

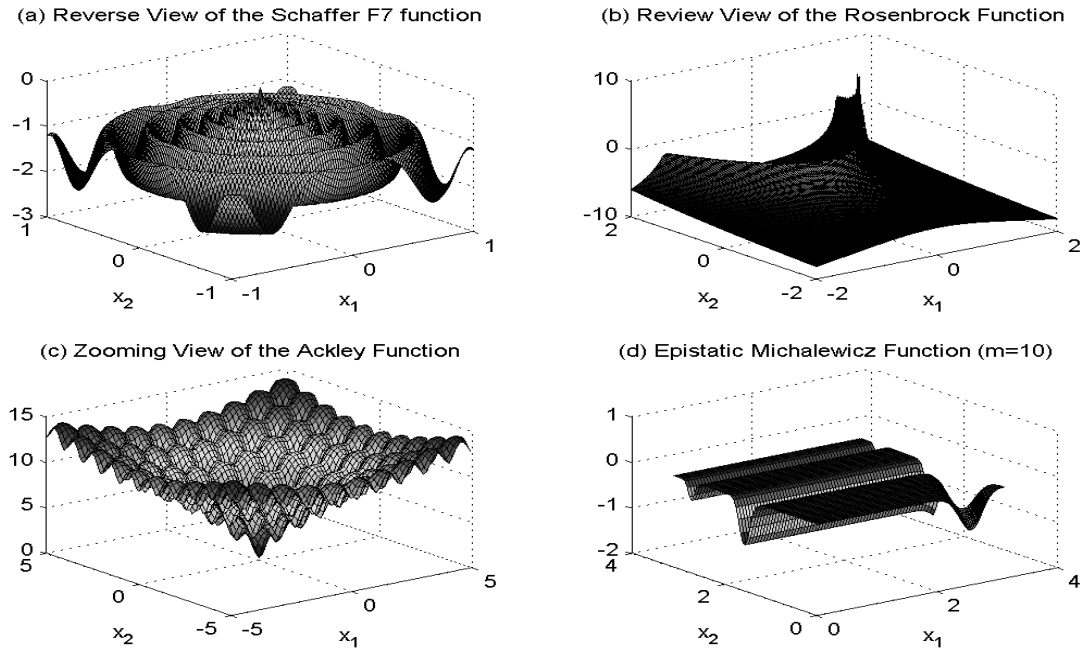

Figure 1: Surface plot for the four testbeds.

the global optimum more quickly and again significantly outperforms the GA in long term performance.

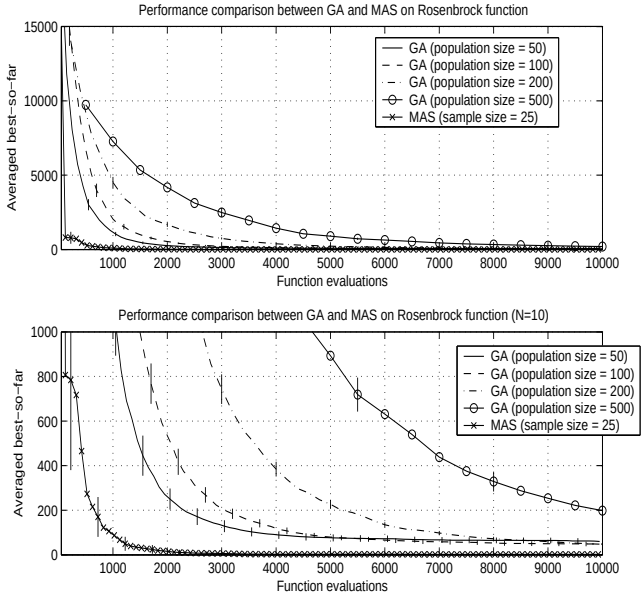

Figure 4: Best-so-far performance on Rosenbrock function.

4.3 The Ackley Path Function

Ackley's Path [1] is a widely used multimodal test function. The function's definition is:

$$f(\overline{x}) = -ae^{-b(\frac{\sum_i x_i^2}{N})^{\frac{1}{2}}} - e^{\frac{\sum_i cos(cx_i)}{N}} + a + e^1,$$

where a=20, b=0.2, $c = 2\pi$, and $-32.768 \leq x_i \leq 32.768$ for $1 \leq i \leq n$.

The problem we use here is again 10 dimensional (N=10); thus the MAS will use ten agents to search the optimum. For the GA, each individual of the population is a 200 bit-string concatenated by ten blocks of 20 bits each encoding a variable. Figure 1.c gives a visual gist of the function in a lower 2-dimensional form. The surface is overall a single deep well with a locally rough surface. For easy viewing of the details the figure is enlarged, showing the region in the neighborhood of the minimum.

The empirical results of the search algorithms on this surface are displayed in Figure 5. It is clear that the PC-based MAS technique again significantly outperforms the GA in early decent towards the minimum.

4.4 The Michalewicz Epistatic Function

The final testbed employed in this section is Michalewicz's epistatic function [14]:

$$f(\overline{x}) = -\sum_{i=1}^{N} sin(y_i) sin^{2m}(\frac{iy_i^2}{\pi}),$$

where

$$\begin{aligned} y_i &= x_i cos\frac{\pi}{6} - x_{i+1} sin\frac{\pi}{6}, \quad \text{if } i \bmod 2 = 1 \text{ and } i \neq N; \\ y_i &= x_{i-1} sin\frac{\pi}{6} + x_i cos\frac{\pi}{6}, \quad \text{if } i \bmod 2 = 0 \text{ and } i \neq N; \\ y_N &= x_N, \end{aligned}$$

$0 \leq x_i \leq \pi$ for $1 \leq i \leq N$.

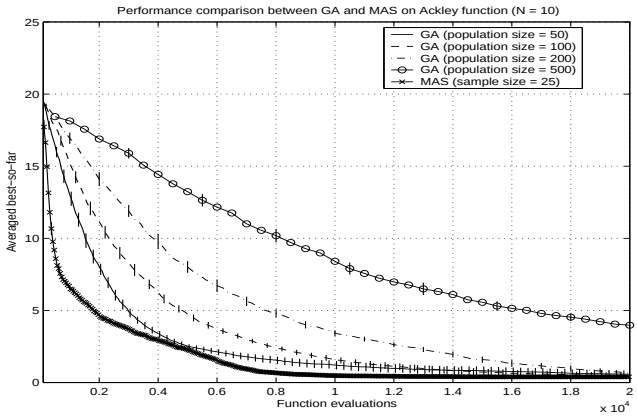

Figure 5: Best-so-far performance on Ackley's Function.

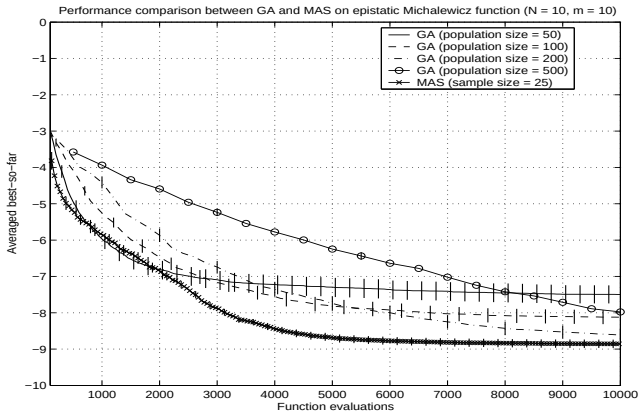

Figure 6: Best-so-far performance on Michalewicz's epistatic function (m=10).

A system is highly epistatic if the optimal allele for any locus depends on a large number of alleles at other loci. The concept of epistasis in nature corresponds to nonlinearity in the context of GA [7]. This function is a highly multimodal, nonlinear and nonseparable testbed ($n!$ local optima). A sketch of a two-dimensional version of this function is displayed in Figure 1.d for the steepness parameter $m = 10$. Larger m leads to more difficult search. For very large m the function behaves like a needle in the haystack since the function values for points in the space outside the narrow peaks give very little information on the location of the global optimum.

The periodic, self-similar valleys can be expected to create more local minima in the product space PD than in the true fully coupled representation. Thus this surface will be hard for PC. Another difficulty comes from the simple rotation of the valleys that couple all the axes in pairs. This explicit coupling is well designed to frustrate the explicitly decoupled Multi-agent probability system. Conversely, because only consecutive alleles are coupled by the rotation, cross-over is well prepared to conserve this coupling in evolution.

We searched a ten dimensional (N=10) space for the cases of $m = 10$ and $m = 200$. As before the GA uses 20 bits per variable. The empirical results are displayed in Figure 6 and Figure 7 for $m = 10$ and $m = 200$, respectively.

It is clear that the PC-based MAS technique again significantly outperforms the GA. In particular, in case of $m = 200$, the PC still demonstrates a surprising search power even though the function behaves like a needle in the haystack and is very difficult to search.

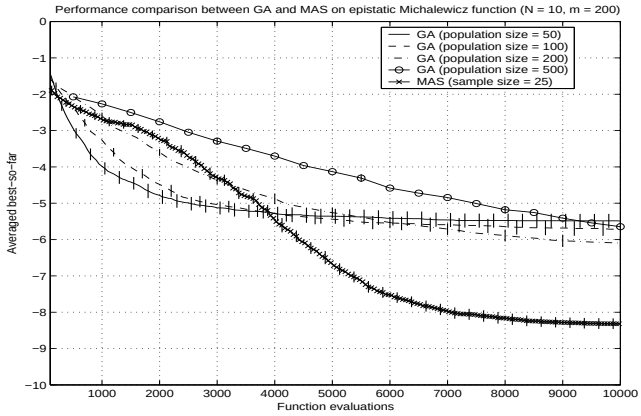

Figure 7: Best-so-far performance on Michalewicz's epistatic function (m=200).

5. DISCUSSION AND CONCLUSION

We presented a comparative study of two agent-based adaptive algorithms – the GA and the PC approach.The PC method appeared superior to the GA method both in initial rate of decent and perhaps more significantly on long term performance for these traditional GA community testbed functions. In comparing the performance curves one should note that the bottom axis is not iterations but is in function evaluations. This is apropos to the common situation where function evaluations are expensive. In the real world, a "function evaluation" might actually require a complex experiment or simulation (such as building an airplane wing shape or a long monte-carlo simulation of a protein folding) or in real-time optimization the sampling rate could be measurement-rate limited. Thus minimizing function calls is desirable for response rate, total speed, and expense. This basis allows one to view how much information is being squeezed out of every precious sample. In the testbed functions just examined the analytic functional forms were, in fact, relatively cheap to evaluate in which case it should be noted that a GA iteration has less overhead than the more complex PD gradient ascent.

The function evaluation axis penalizes methods that take more samples per iteration without proportionally improving their performance; an advantage of the PC method is it performs quite well with fewer samples per iteration than the GAs. We note that while the smallest GA population sizes have faster initial descent rates than their larger brethren, they typically converged to sub-optimal solutions away from the global minimum so that decreasing the population size of the GA to match the PC performance would have worsened long term performance. Additionally, the PC best-so-far trajectory was far more reproducible than that of the

low-population GAs.

The PC approach introduces a methodology by which the search course in this system is guided by probability distribution over variables, rather than using single values derived from those variables. The resulting distributed algorithms can facilitate the search for robust and scalable solutions to difficult problems. Using several examples of parametric optimization problems we thus demonstrate the power of this PC-based MAS methodology for estimation of distributed algorithms, which can significantly outperform the traditional GA on complex function optimization problems.

6. REFERENCES

[1] D. H. Ackley. *A connectionist machine for genetic hillclimbing*. Kluwer Academic Publishers, Boston, 1987.

[2] S. Baluja and S. Davies. Combining multiple optimization runs with optimal dependency trees. Technical Report CMU-CS-97-157, Carnegie Mellon University, 1997.

[3] S. Bieniawski, D. H. Wolpert, and I. Kroo. Discrete, continuous, and constrained optimization using collectives. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA Paper 2004-4580*, 2004.

[4] J. De Bonet, C. Isbell Jr., and P. Viola. Mimic: Finding optima by estimating probability densities. In *Advances in Neural Information Processing Systems - 9*. MIT Press, 1997.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd ed.)*. Wiley and Sons, 2000.

[6] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991.

[7] D. E. Goldberg. *Genetic Algorithms in search, Optimization, and Machine Learning*. Addison Wesley, Reading, MA, 1989.

[8] D. E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. *Foundation of Genetic Algorithms*, pages 69–93, 1991.

[9] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[10] ICEO. Second international contest on evolutionary optimization, held in the ieee-icec 97 conference. 1997.

[11] P. Larranga and J. A. Lozano, editors. *Estimation of Distribution Algorithms. A new Tool for Evolutionary Computation*. Kluwer Academic Publishers, 2002.

[12] C. Lee and D. Wolpert. Product distribution theory for control of multi-agent systems. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, pages 522–529. IEEE Press, 2004.

[13] D. Mackay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[14] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[15] R. Rubenstein. The stochastic minimum cross-entropy method for combinatorial optimization and rare-event estimation. unpublished, 2005.

[16] R. Rubinstein and D. Kroese. *The Cross-Entropy Method*. Springer, 2004.

[17] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ, 1995.

[18] P. Sabes and M. Jordan. Reinforcement learning by probability matching. In *Advances in Neural Information Processing Systems - 8*. MIT Press, 1995.

[19] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das. A study of control parameters affecting online performance of genetic algorithms for function optimization. In *Proc. 3rd International Conference on Genetic Algorithms*, pages 51–60. Morgan Kaufmann, 1989.

[20] K. Tumer and D. Wolpert. Collective intelligence and braess' paradox. In *AAAI/IAAI*, pages 104–109, 2000.

[21] D. Wolpert and S. Bieniawski. Distributed control by lagrangian steepest descent. In *Proceeding of the Conference of Decision and Control*, 2004.

[22] D. Wolpert and S. Bieniawski. Product distributions, bias plus variance, and collective intelligence. In *Proceedings of WEHIA 04*. Springer Verlag, 2004.

[23] D. H. Wolpert. Theory of collective intelligence. In K. Tumer and D. H. Wolpert, editors, *Collectives and the Design of Complex Systems*, New York, 2003. Springer.

[24] D. H. Wolpert. Theory of collectives. In *The Design and Analysis of Collectives*. Springer-Verlag, New York, 2003. Available at http://ic.arc.nasa.gov/ dhw.

[25] D. H. Wolpert. Bounded rational games, information theory, and statistical physics. *In D. Braha and Y. Bar-Yam, editors, Complex Engineering Systems*, 2004.

[26] D. H. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2/3):265–279, 2001.

[27] D. H. Wolpert, K. Tumer, and E. Bandari. Improving search by using intelligent coordinates. *Physical Review E*, 2003. In press.

[28] D. H. Wolpert, K. Tumer, and J. Frank. Using collective intelligence to route internet traffic. In *Advances in Neural Information Processing Systems - 11*, pages 952–958. MIT Press, 1999.

[29] D. H. Wolpert, K. Wheeler, and K. Tumer. Collective intelligence for control of distributed dynamical systems. *Europhysics Letters*, 49(6), March 2000.